Applying ML Techniques to Classify HEP Collision Data

Sumit Kumar Adhya : 23b1806

Instructor : Prof. Alok Shukla

Course Project | AI-DS

OVERVIEW

Some Terminologies

1 Diffractive events in high-energy physics are characterized by the exchange of a colorless object, such as a Pomeron, in the t-channel, leading to a large rapidity gap (a region with no particles) between the scattered particles and the rest of the event. These events often involve elastic scattering or proton dissociation. [1]

2 Non-diffractive events, on the other hand, do not exhibit such rapidity gaps and involve more uniform particle production across the rapidity range. [2] These events are typically associated with inelastic collisions, where the interacting particles lose energy and produce secondary particles. [3]

3 Charged particle multiplicity distribution refers to the probability distribution of the number of charged particles produced in a high-energy collision event. [4] It is a fundamental observable that characterizes the final state of the collision. The distribution provides insights into the underlying particle production mechanisms and the dynamics of the collision process. [5]

Problem Statement

The charged particle multiplicity distribution in High Energy Physics receive contribution from both Diffractive and Non-Diffractive processes. It is experimentally challenging to segregate these two kinds of events. The aim of the project is to apply different Machine Learning algorithms on High Energy collision data to classify events into diffractive and non-diffractive ones. The codes and datasets used in this project can be found in the link below : link

Approach

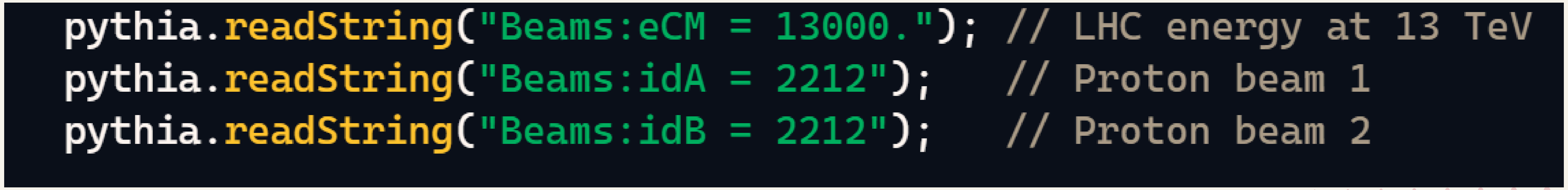
There are a lot of datasets related to HEP online . But we would not use them. We would rather curate a realistic dataset by ourselves. This is because most of the datasets online are much more complicated and don't contain the information we need and therefore would not solve our purpose. We would rather use an event generating software named Pythia 8, which is a popular simulating tool in Computational High Energy Physics. We would then use various machine learning algorithms for classification.

Pythia 8

PYTHIA is a program for the generation of high-energy physics collision events. It contains theory and models for a number of physics aspects, including hard and soft interactions, parton distributions, initial- and final-state parton showers, multiparton interactions, fragmentation and decay. It is largely based on original research, but also borrows many formulae and other knowledge from the literature. As such it is categorized as a general-purpose Monte Carlo event generator. [6]

Curating a realistic dataset

- Pythia 8 runs in a Unix environment and uses C++ for it's functioning. The code given the following link has been used to create our dataset.
- Inside main, the following settings has been used to simulate two colliding proton beams at an energy of 13 TeV (typically used at LHC)



```
  pythia.readString("Beams:eCM = 13000."); // LHC energy at 13 TeV
  pythia.readString("Beams:idA = 2212");   // Proton beam 1
  pythia.readString("Beams:idB = 2212");   // Proton beam 2
```

- Two output csv files (for diffractive and non-diffractive events) are then chosen to store the data of the particles emitted after collision of these two beams.
- The following features are then extracted for each particle emitted during the collisions.

Features

Transverse momentum (pT)

The momentum of a particle in the plane transverse (perpendicular) to the beam axis.

Pseudo-rapidity

A measure of the angle of the emitted particle relative to the beam axis (z-axis).
$\eta=-\ln[\tan(\theta/2)]$

Azimuthal Angle (ϕ)

The azimuthal angle of a particle in the transverse plane, measured around the beam axis. It's typically expressed in radians: $\phi\in[0,2\pi)$

Rapidity

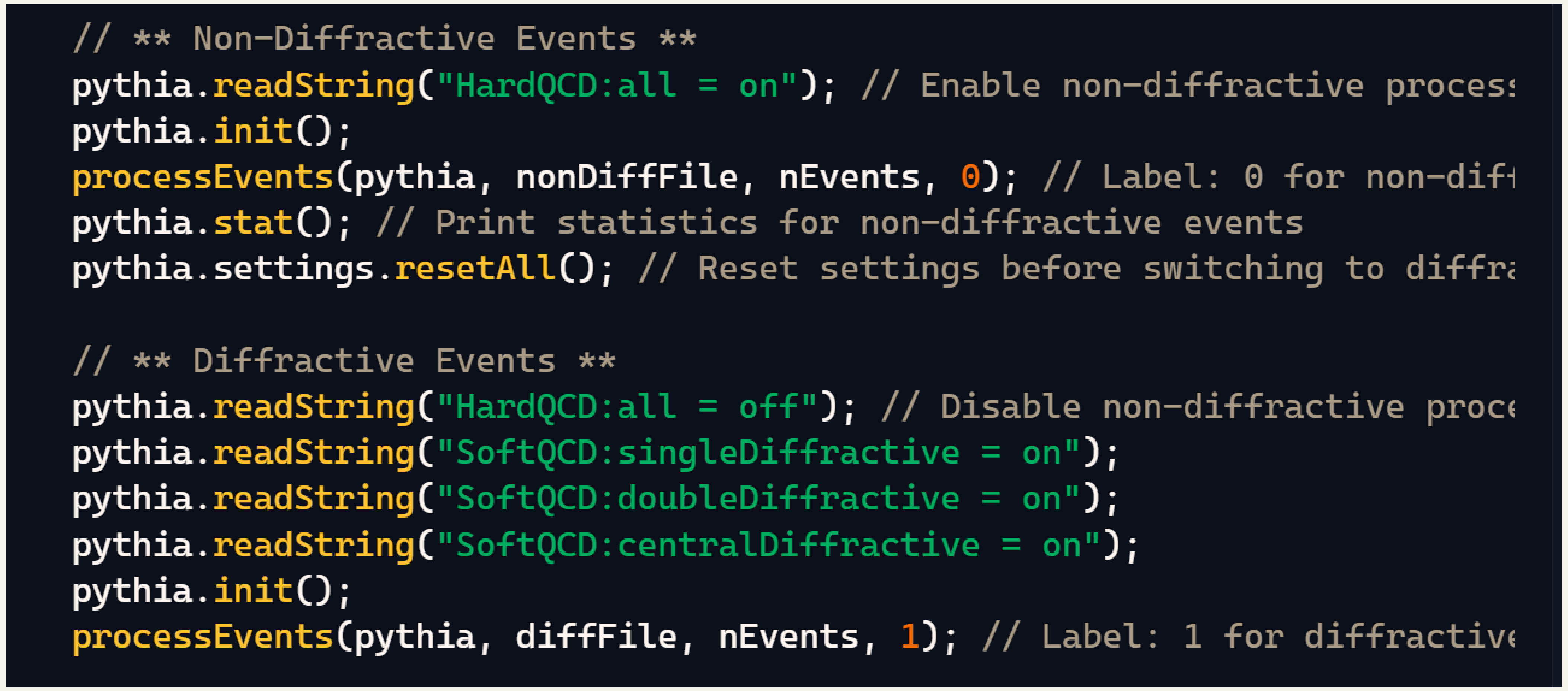
Rapidity is a relativistic measure of motion along the beam axis.
$y=1/2\ln(E-pz/E+pz)$

Features

There are other features like 'Event' which denotes the serial no. of events. Each event produces many particles, the ids of which are stored in the 'ParticleID' feature. Whether a particle is charged or not is denoted by 1 or 0 under the 'Charged' feature. The energies of every particle is stored under the 'Energy' feature and finally the Class i.e. Diffractive : 1 and Non-Diffractive : 0 in the Class variable.

Curating a realistic dataset

- The code simulates 5000 collisions (events) each for diffractive and non-diffractive type.
- The following settings are used to generate diffractive and non-diffractive events.



```
// ** Non-Diffractive Events **
pythia.readString("HardQCD:all = on"); // Enable non-diffractive proces
pythia.init();
processEvents(pythia, nonDiffFile, nEvents, 0); // Label: 0 for non-dif
pythia.stat(); // Print statistics for non-diffractive events
pythia.settings.resetAll(); // Reset settings before switching to diffr

// ** Diffractive Events **
pythia.readString("HardQCD:all = off"); // Disable non-diffractive proce
pythia.readString("SoftQCD:singleDiffractive = on");
pythia.readString("SoftQCD:doubleDiffractive = on");
pythia.readString("SoftQCD:centralDiffractive = on");
pythia.init();
processEvents(pythia, diffFile, nEvents, 1); // Label: 1 for diffractive
```

- The features are then extracted by the processEvents function. The rapidities are calculated and appended by the calculateRapidity function

Data Preprocessing and Feature Extraction

- The following python code has been used to study this data: link
- The data is first loaded and checked for nan and inf values.
- The following features are extracted for analysis. The raw data generated using Pythia contains information for each particle emitted in every event. However, for classifying an event as diffractive or non-diffractive, the aggregate properties of the entire event are found to be more significant than the individual particle data. The DataFrame.groupby feature of pandas has been used therefore to accomplish the same.
- The Total multiplicity : Total no. of particles generated in each event & Charged multiplicity : Total no. of charged particles generated in each event and Charge Ratio : Charged multiplicity/Total multiplicity are calculated.

Data Preprocessing and Feature Extraction

- The mean and var of various features in each event is calculated. Two more features are added; Total Energy : Total energy of all the particles emitted in a particular event and Energy Ratio : Max Energy in an event / Total Energy.
- A new feature Angular Correlation : The correlation between the azimuthal angle (ϕ) and the pseudorapidity (η) for all particles in a given event is calculated. We are doing this because a strong correlation (positive or negative) may suggest an event-specific angular pattern, while a weak correlation indicates a more random angular spread.
- So the above features are calculated using the groupby feature of pandas for both diffractive and non-diffractive dataframes. They are finally combined and randomized to create our final processed data (10,000 rows * 14 columns).

Feature Selection

- As can be seen from the processed data, the values of the Total Energy feature is almost the same for all the events ≈ 13TeV. This is very obvious by the Law of Conservation of Energy. Energy of initial beam = Total energy of emitted particles. So we drop out this feature.

- We then used the following tools to figure out the features important for our study.
    - Correlation matrix using Seaborn : Helps detect linear relationships between features. It identifies redundancy and potential multicollinearity, which might affect models like linear regression. Correlation doesn't capture non-linear relationships.

Feature Selection

- **Using XGB Classifier : Inherently handle non-linear relationships and interaction effects between features. They rank features based on their ability to improve the split decision and reduce impurity.**

- **Recursive Feature Elimination (RFE) : Evaluates feature importance by iteratively building models and removing the least important feature at each step.**

- **Shapley Additive Explanations (SHAP) : Provides more granular insight into how each feature influences the output of complex models, like tree-based models. SHAP values provide both the magnitude and direction of feature influence.**

Feature Selection

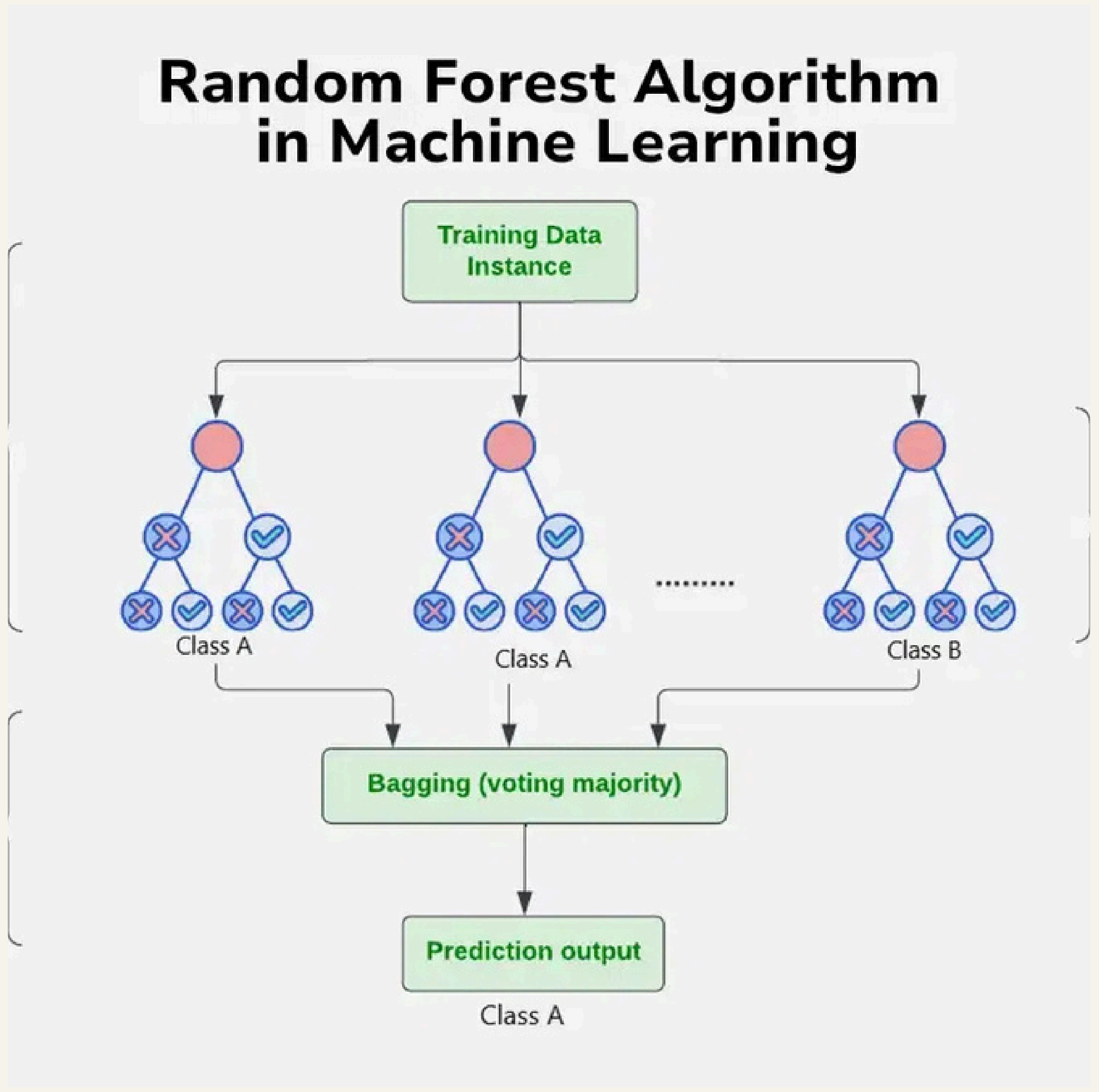
The reason why we are using different tools for feature selection rather than just one is because each method has its strengths and weaknesses, and they capture different aspects of the data. It also reduces the risk of overfitting by avoiding reliance on a single tool and enhances model interpretation by providing confidence in features that are consistently identified as important across methods, while also helping to discard redundant or irrelevant features, improving dimensionality and interpretability.

Analyzing the correlation matrix, it turns out that the pairs Total Multiplicity-Charged Multiplicity, Mean_Eta-Mean_Rapidity and the Var_Eta-Var_Rapidity pairs are highly correlated. The first one is very obvious to be so while the last two happen because of the following physical reason:

Feature Selection

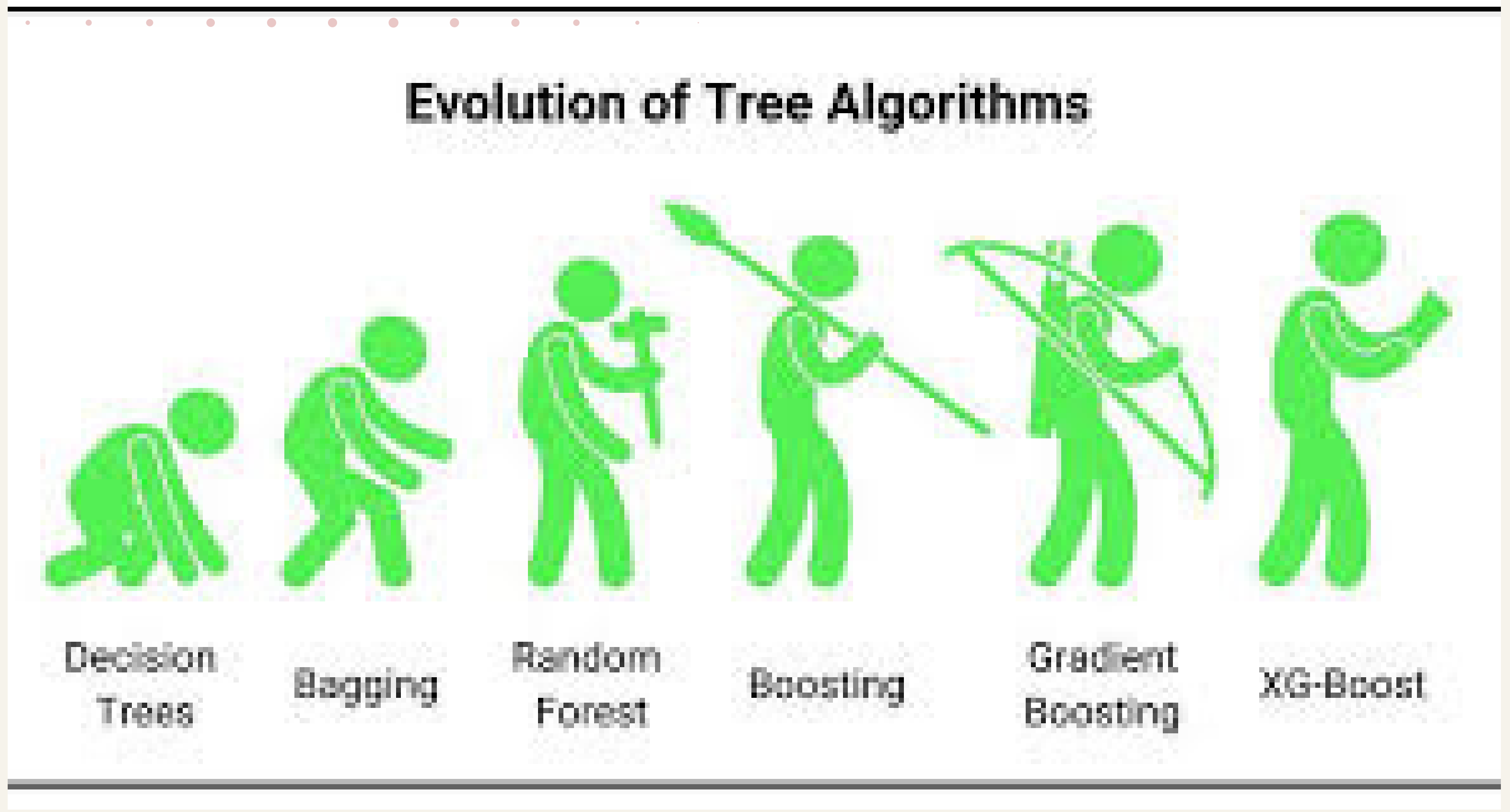
In the relativistic limit (for high-energy particles), rapidity and pseudorapidity become approximately equal as the particle's transverse momentum pT is much smaller compared to its longitudinal momentum pz. This can also be confirmed by looking at the nearly equal values of Rapidity and eta from the original data. So we are going to drop one of these from our data. Although Total and Charged multiplicity seem to be correlated but they might have a combined effect on the Class variable. In any way, since we are not going to use any linear model for our study, let’s turn to the other tools we have which takes non linearity into account. Based on the feature importances given by these other tools, we finally decide the following features to be important : ‘Total Multiplicity’, ‘Charged Multiplicity’, ‘Mean_Eta’, Var_Rapidity’ and ‘Energy Ratio’. Now we use different ML algorithms for classification.

Random Forest


Random Forest Algorithm in Machine Learning
Training Data Instance
Class A
Class A
Class B
Bagging (voting majority)
Prediction output
Class A

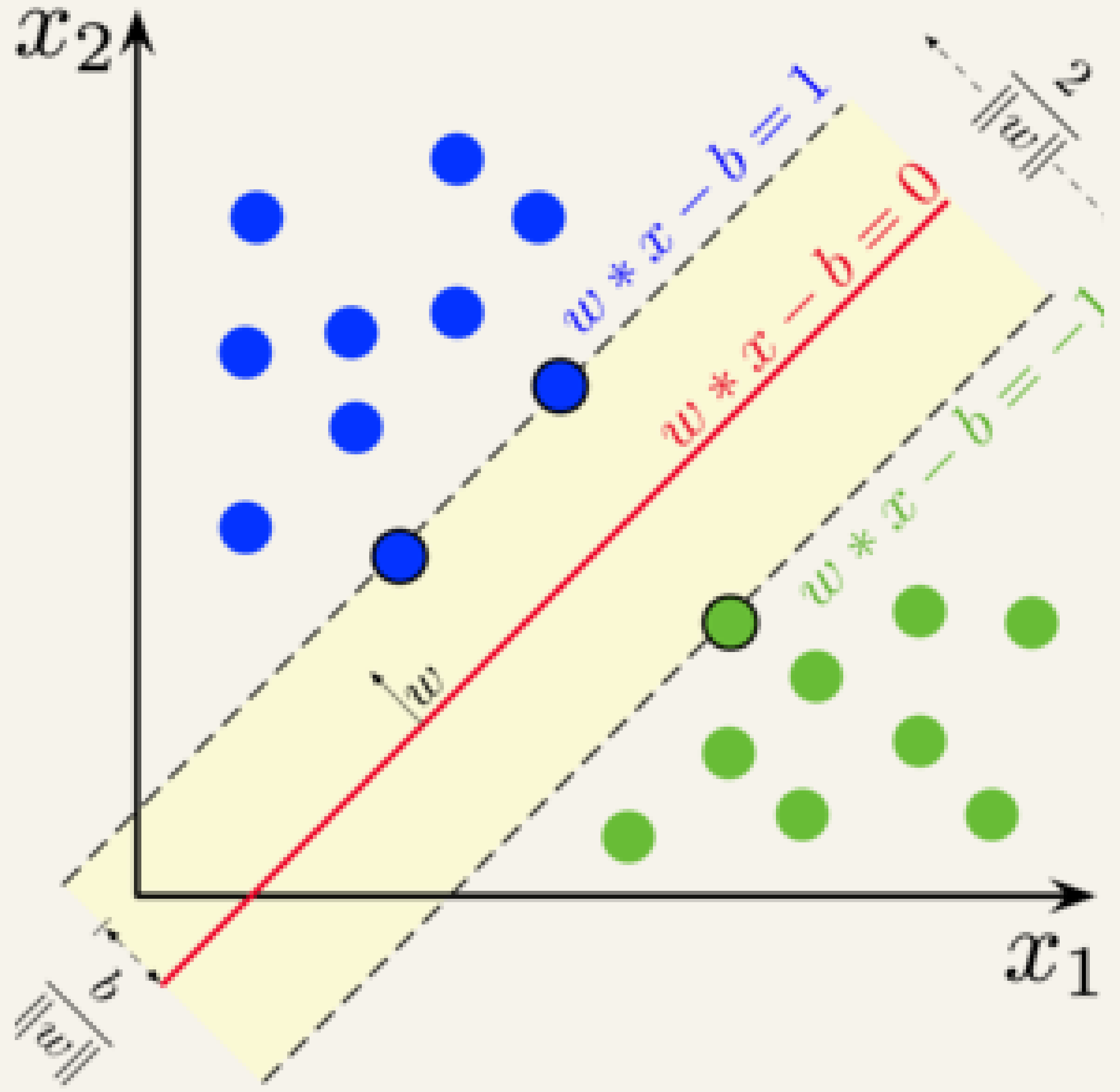
Random Forest is an ensemble learning algorithm primarily used for classification and regression tasks. It builds multiple decision trees during training and outputs the majority vote (for classification). In high-energy physics (HEP), Random Forests are widely used for tasks such as particle identification, event classification, and signal-background discrimination. Their ability to handle high-dimensional, noisy data makes them particularly valuable in tasks like ours. [7]

XGBoost



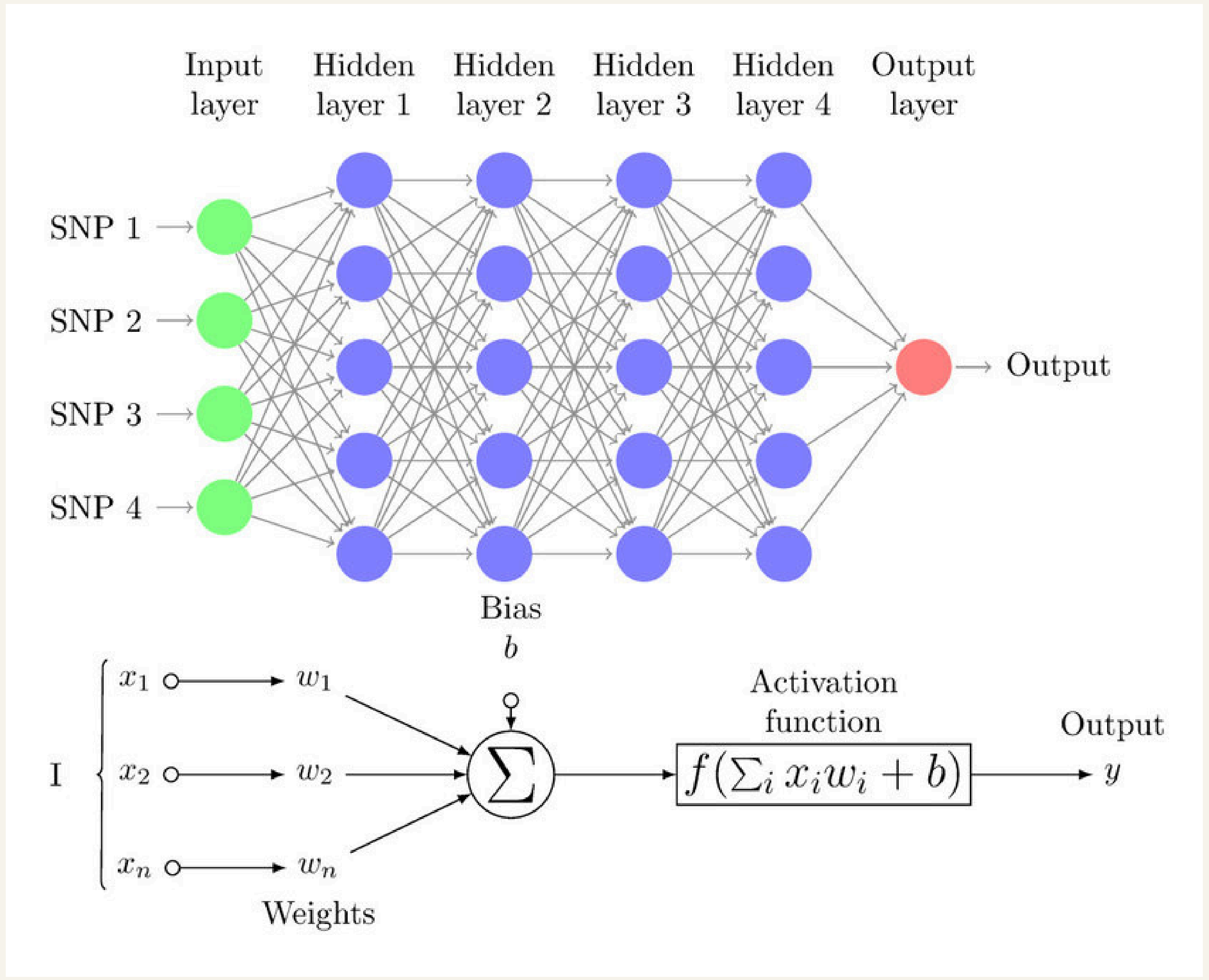
XGBoost (Extreme Gradient Boosting) is a powerful machine learning algorithm that uses gradient boosting frameworks to optimize predictive models. It builds multiple decision trees sequentially, with each tree correcting the errors of its predecessor, resulting in a highly accurate model. In high-energy physics, XGBoost is used for various applications, such as particle identification, event classification, and jet tagging. For instance, it has been employed to distinguish between signal and background events in searches for new physics at the Large Hadron Collider (LHC). [8]

Support Vector Machines



A Support Vector Machine (SVM) is a supervised machine learning algorithm that works by finding the hyperplane that best separates the data points in a high-dimensional space. The data points that are closest to the hyperplane (the support vectors) determine the position and orientation of the hyperplane. In high-energy physics, SVMs are used for various applications, like particle and event identification or Identifying potential new physics phenomena, such as searches for Supersymmetry at the LHC.

Multilayer Perceptron



A Multilayer Perceptron (MLP) is a type of artificial neural network that consists of at least three layers of nodes: an input layer, one or more hidden layers, and an output layer. Each node, or neuron, in one layer connects with a certain weight to every node in the following layer, and these weights are adjusted during training to minimize the error in predictions.

MLPs are valued for their ability to model complex, non-linear relationships in data, making them suitable for the intricate patterns found in high-energy physics data.

Implementation

All the algorithms mentioned have been applied on our processed and filtered data and their accuracies and classification reports have been printed as can be seen in our code

		Accuracy
• Random Forest	-	91.55%
• XGBoost	-	91.85%
• SVM	-	92%
• MLP	-	91.75%

Introducing a new feature : Rapidity Gaps

This new feature we are going to introduce in our code will appear to be a magic feature.

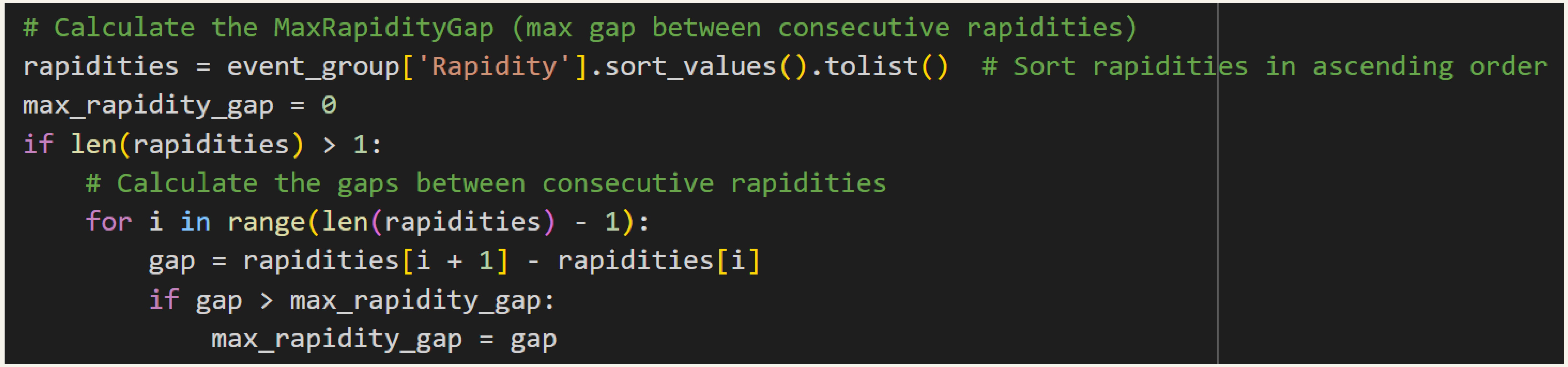
It is an important feature in classifying diffractive and non-diffractive events because they provide a clear signature that helps distinguish between these two types of events. In diffractive events, the exchange of a colorless object, such as a Pomeron, leads to a suppression of particle production in certain regions of the detector, creating a rapidity gap—a region devoid of particlesl. This absence of particles in the rapidity gap is a distinctive signature of diffractive events. [9]

Introducing a new feature : Rapidity Gaps

In contrast, non-diffractive events do not exhibit such rapidity gaps, as particles are produced more uniformly across the rapidity range. By measuring and analyzing the presence or absence of rapidity gaps, researchers can effectively classify events and study the underlying physics processes. [9]
The reason why we didn't use this feature till now is because we wanted to test what other features are important in determining whether an event is diffractive or non-diffractive. Now that we have obtained a 92% accuracy without this, let's give a try on this as well.

Introducing a new feature : Rapidity Gaps

We use the following algorithm to calculate the maximum rapidity gap in each event.



```python
# Calculate the MaxRapidityGap (max gap between consecutive rapidities)
rapidities = event_group['Rapidity'].sort_values().tolist()  # Sort rapidities in ascending order
max_rapidity_gap = 0
if len(rapidities) > 1:
    # Calculate the gaps between consecutive rapidities
    for i in range(len(rapidities) - 1):
        gap = rapidities[i + 1] - rapidities[i]
        if gap > max_rapidity_gap:
            max_rapidity_gap = gap
```

Introducing a new feature : Rapidity Gaps

We then perform data preprocessing and feature selection on our new data. Then we use the XGBoost Classifier for classification.

As can be seen in the code the feature importance of this new feature comes out to be much higher than others, so much so that the machine when trained on this single feature alone gives a whooping 94.05% accuracy. It is no wonder though. It is rather very much expected according to the physics discussed before.

Result and Conclusion

Throughout this project, we have successfully demonstrated the application of various Machine Learning (ML) algorithms to classify High Energy Physics (HEP) collision events into diffractive and non-diffractive categories. By leveraging Pythia 8, we simulated realistic event data and carefully processed it to derive meaningful features that capture the underlying physics.

We applied several ML algorithms, including Random Forest, XGBoost, Support Vector Machines (SVM), and Multilayer Perceptron (MLP), to evaluate their performance on the classification task. While all models achieved comparable accuracies, the SVM emerged as the best performer, achieving an impressive accuracy of 92%.

Result and Conclusion

Among the features derived, we introduced a novel feature, Rapidity Gap, which plays a crucial role in distinguishing diffractive events. Remarkably, training the model using only this single feature yielded an exceptional accuracy of 94.05%, outperforming models trained on the entire feature set. This highlights the profound physical relevance of the rapidity gap in the classification of HEP events.

Through this study, we have not only demonstrated the potential of ML in analyzing HEP data but also provided insights into feature importance and their correlation with physical phenomena.

Bibliography

1. https://arxiv.org/abs/1206.2124?form=MG0AV3
2. https://arxiv.org/abs/2206.03199?form=MG0AV3
3. https://cds.cern.ch/record/1176140/files/Pages_from_C09-03-14--1_281.pdf?form=MG0AV3
4. https://dspace.mit.edu/bitstream/handle/1721.1/133891/Khachatryan2011_Article_ChargedParticleMultiplicities1.pdf?sequence=2&form=MG0AV3
5. https://link.springer.com/chapter/10.1007/978-981-15-6292-1_25?form=MG0AV3
6. https://pythia.org

Bibliography

7.https://link.springer.com/article/10.1007/s10915-023-02144-2?form=MG0AV3
8.https://link.springer.com/article/10.1140/epjp/s13360-024-05781-0?form=MG0AV3
9.https://cds.cern.ch/record/1641424/files/ATL-PHYS-PROC-2014-004.pdf?form=MG0AV3

THANK YOU